A Benders Decomposition-Based Matheuristic for the Cardinality Constrained Shift Design Problem

Lusby, Richard Martin ; Range, Troels Martin; Larsen, Jesper

Published in:
European Journal of Operational Research

Link to article, DOI:
10.1016/j.ejor.2016.04.014

Publication date:
2016

Document Version
Peer reviewed version

A Benders decomposition-based Matheuristic for the Cardinality Constrained Shift Design Problem

Richard Martin Lusby[a,*], Troels Martin Range[b], Jesper Larsen[a]

[a] *Department of Management Engineering, Technical University of Denmark*
[b] *Department of Business and Economics, University of Southern Denmark*

Abstract

The Shift Design Problem is an important optimization problem which arises when scheduling personnel in industries that require continuous operation. Based on the forecast, required staffing levels for a set of time periods, a set of shift types that best covers the demand must be determined. A shift type is a consecutive sequence of time periods that adheres to legal and union rules and can be assigned to an employee on any day. In this paper we introduce the Cardinality Constrained Shift Design Problem; a variant of the Shift Design Problem in which the number of permitted shift types is bounded by an upper limit. We present an Integer Programming model for this problem and show that its structure lends itself very naturally to Benders decomposition. Due to convergence issues with a conventional implementation, we propose a matheuristic based on Benders decomposition for solving the problem. Furthermore, we argue that an important step in this approach is finding dual alternative optimal solutions to the Benders subproblems and describe an approach to obtain a diverse set of these. Numerical tests show that the described methodology significantly outperforms a commercial Mixed Integer Programming solver on instances with 1241 different shift types and remains competitive for larger cases with 2145 shift types. On all classes of problems the heuristic is able to quickly find good solutions.

Keywords: Scheduling, Integer programming, Shift design, Benders decomposition

1. Introduction

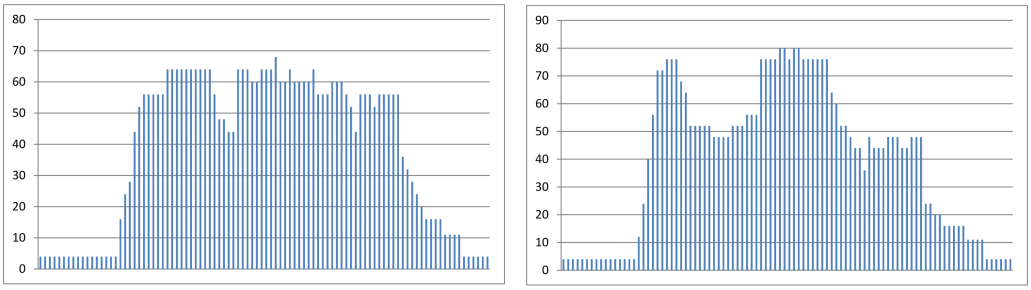
Satisfying the demand for service is a key objective for many companies, and employees are typically the main resource available to achieve this. While demand may fluctuate during the course of a day, each staff member is usually assigned to a shift, which starts at a particular time of the day and lasts a certain duration, making it difficult to match the demand for employees with the actual number of employees at work. Using different types of shifts makes it possible to better match the demand. This type of decision problem mainly arises in industries that require continuous operation, examples of which include airlines, airports, police and emergency services, financial institutions, transportation, and call centers.

The process of personnel scheduling is often divided into several steps that are carried out sequentially. Initially, the staffing level required in each of a sequence of consecutive time

*Corresponding author, *ph: +45 4525 3084*
Email address: rmlu@dtu.dk (Richard Martin Lusby)

periods is determined. The duration of all time periods is often the same and is typically short, e.g. from five to 15 minutes. Based on union rules and legal requirements shifts are then established. A shift is a piece of work that spans multiple time periods and is often equal to a daily piece of work for an employee. Whereas a shift is a particular time interval on a given day, a shift type is a time interval on any day. An example of a shift type could be a *morning shift* that is used every day from 4am to 11am. The set of all feasible shifts can be extremely large. Therefore, after establishing the demand scenario(s), i.e. a daily forecast of the number of employees required for specific intervals of time, one needs to solve a *Shift Design Problem (SDP)*. This entails determining a cost-effective set of shifts that (in general) cover the demand. In the final personnel rostering phase, named employees are assigned to a sequence of shifts covering the full rostering period (typically a month, three months or half a year). This final phase also contains the assignment of days-off, rest periods, availablities and preferences.

The authors were exposed to a challenging application of the SDP when working with personnel scheduling for the main security area at Copenhagen Airport, see Nielsen (2012). Here the staffing level required has weekly cycles. Therefore, when scheduling the personnel the demand scenario for each of the seven days must be taken into account, as there are daily differences. For instance, all Mondays look quite similar. As an example Figure 1 gives the required staffing levels in the main security area for a Monday and a Saturday, respectively. From the figures the challenge of the SDP is clear; the set of shift types permitted on each day must be the same, despite the obvious difference in demand scenarios.



(a) Monday (b) Saturday

Figure 1: Example demand profiles

In order to handle the complexity of the rostering process a total of 16 shift types are used throughout the week. That is, a total of 16 shift types are available for ensuring a proper coverage of the demand on any of the seven days in the week. Needless to say, compromises must be made between the demand coverage on any given day and the solution quality since the shift types must encapsulate the most important structures of each day of the week. In fact, analysis shows that solving each day to optimality independently requires between 37 and 50 shifts, and that although there are shifts that appear in the optimal solution of more than one day, there is also a great variation in the chosen shifts, see Nielsen (2012). What is therefore very interesting for airport management, and also for SDPs in general, is to work with a constrained version of the problem. In the so-called *Cardinality Constrained Shift Design Problem (CCSDP)* one needs to determine the most cost-effective set of shift types given a strict upper bound on the number of shift types allowed.

In contrast to SDPs found in the literature, this application has a strict cardinality constraint on the number of shift types which are allowed across all demand scenarios. This makes the problem significantly harder to solve, as one set of shift types may be good for one scenario, but very bad for another. Hence a compromise is necessary to minimize the overall deviation from staff demand. Like other SDPs, we allow for over and undercoverage, i.e. it is not a strict requirement that demand is met exactly, but penalties are incurred for having too many (or too few) staff in a given time period.

We do, however, make two important assumptions regarding the nature of the underlying problem. Firstly, we do not explicitly model employee breaks; we assume that a shift type covers a consecutive sequence of periods, the duration of which includes any necessary break time. The exact timing of an employee's break is decided by the supervisor on duty when the employee is at work. This is consistent with what is done in practice at Copenhagen Airport, and allows the supervisor to dynamically modify the break patterns to the workload on a specific day. Secondly, we remove from consideration those shift types that span consecutive days, e.g. a night shift that starts on Friday at 10pm and finishes on Saturday at 6am. Typically security personnel must be required in such periods, and the number required does not vary significantly day-to-day. We assume the demand scenarios account for this. Furthermore, this allows a daily decomposition approach to be considered.

To solve the CCSDP we propose a Benders decomposition based matheuristic that only ever considers a small set of shift types at any one time. We observe that it is particularly easy to solve the problem when the set of used shifts is given up front. Through an analysis of the Benders cuts found while solving a reduced problem, the set of considered shift types is dynamically updated. We highlight the importance of finding dual alternative optimal solutions to the Benders subproblem when determining which shift types to include in set and describe a procedure for finding a diverse set of such solutions. Numerical tests show that the described methodology significantly outperforms a commercial Mixed Integer Programming solver on instances with 1241 different shift types and remains competitive for larger cases with 2145 shift types. On all classes of problems the heuristic is able to quickly find good solutions.

The paper unfolds in the following way. First, Section 2 provides a review of the related literature. This is followed in Section 3 by a formal description of the model. Here we also argue that a specific special case of the SDP is easy to solve. In Section 4, this model is reformulated using Benders decomposition. We discuss how to derive several different Benders cuts. The model described includes a significant number of big-M constraints and is therefore hard to solve. Instead of solving the model to optimality we describe a matheuristic based on Benders decomposition in Section 5. Section 6 provides a description and the analysis of computational experiments for the heuristic. A brief discussion on extensions is given in Section 7, while Section 8 gives some concluding remarks.

2. Related Literature

Personnel scheduling is an important and classical problem in Operations Research and has been widely studied. The survey Ernst et al. (2004b) gives an extensive review on personnel scheduling. It contains a classification of methods, models and problem types collected from almost 200 citations. In addition, the same authors have published an annotated bibliography Ernst et al. (2004a), containing a rich set of notes covering 700 individual papers. The most updated review on the area is Van den Bergh et al. (2013).

The flexibility of the SDP is determined by shift length, start time and break placements. A set covering approach for this problem was originally developed by Dantzig (1954). The approach basically enumerates all possible shifts by treating every feasible combination of parameters as a new shift. The fundamental problem with this is the rapid increase in the number of decision variables. Many subsequent developments are based on this approach.

Break placement is part of the SDP and considers the important problem of placing the breaks within a shift. Break placement is very often an influential factor in making the set covering approach intractable and has therefore been investigated in several papers. An integer programming approach is proposed in Gaballa and Pearce (1979). Flexibility in break placement is incorporated by including a separate variable for every feasible break option allowed in a shift. This can produce an extremely large number of variables, and therefore an alternative break placement strategy was proposed by Bechtold and Jacobs (1990). In Bechtold and Jacobs (1990), a new set of variables is introduced; each variable represents the total number of employees on all shifts starting their break at a particular time. Thompson (1990) also describes a set covering approach to match employees to shifts and also includes the scheduling of meal breaks. The linear program is used to generate shifts in a construction heuristic. This paper also introduces under and overcoverage to the model of Dantzig (1954), and has an objective function that sums the cost of under and overcoverage, like the problem we consider.

A new integer programming formulation allowing multiple breaks and break windows is developed in Aykin (1996). Breaks are only modelled implicitly and the resulting formulation therefore has a much smaller size than the classical set covering model for the same problem. A comparison of some of the different modelling approaches can be found in Aykin (2000). Computational tests show that the model based on Aykin (1996) is superior to the approach suggested in Bechtold and Jacobs (1990).

With respect to solution methods applied in conjunction with modelling, a variety of heuristic and exact methods have been implemented over the years. To give a few examples, Thompson (1996) implements a simulated annealing heuristic for the problem, Musliu et al. (2004) is based on local search and five different kinds of move operators that iteratively change the incumbent, whereas Glover et al. (1983) is based on artificial intelligence. Exact solution approaches for large-scale problems are based on branch-and-cut, see e.g. Aykin (1998), and branch-and-price, see e.g. Mehrotra et al. (2000).

As the first papers only focused on minimizing the use of staff, the developed models did not allow for undercover. To the authors' knowledge, the first to consider undercover was Butler and Maydell (1979). Here the authors consider a staff rostering problem for the Edmonton police department and attempt to roster staff in a way that minimised the deviation from the desired staffing levels, permitting under as well as overcover. One of the first contributions that explicitly mentions minimizing the number of shifts as part of the objective is Musliu et al. (2004). The paper argues for the use of heuristics and describes a local search approach that minimizes the number of shifts active, while minimizing under and overcover as well as the average number of duties per week. Furthermore, in Di Gaspero et al. (2007) the objective function is almost identical to Musliu et al. (2004), i.e. it includes an aspect minimizing the number of shifts. The paper describes a slightly more theoretical variant of the problem and terms it the minimum shift design problem. The authors show that the problem can be solved using a minimum edge-cost flow problem and also present some hardness properties in the paper. Finally, based on the hardness of the problem they suggest a local search method heavily inspired by Musliu et al. (2004).

In the seminal paper Dantzig (1954) no instances nor experiments are given. The method

is explained using an example containing six time periods and six patterns. In the following literature instances are created either randomly or based on practical problems. The papers Thompson (1990, 1996) look at different demand profiles with a different number of demand peaks. For other papers it is often only one characteristic based on the problem at hand (e.g. call centre or airport). In Thompson (1990, 1996), operating days of 15 hours and 20 hours are considered. Using 15 minute intervals and breaks this leads to instances of up to 6600 shifts. Similarly, based on 24 hour a day service and 15 minute intervals, Aykin (1996) reports instances with up to 8640 shifts. In Mason et al. (1998) a weekly problem for an airport is solved. Using 80 15 minute time periods a day this leads to 195 full-time shifts being used in the optimal solution. This corresponds to 39 full-time staff for a weekly schedule (five days-on, two days-off). Finally, Eveborn and Rönnqvist (2004) present a general scheduling method based on column generation. Several scenarios from different industries are solved (retail and call centres). In the largest instance 200 persons must be scheduled, and the instances span between one and three weeks, leading to between 140 and 448 time periods.

3. A model for the CCSDP

In this section we formulate the CCSDP as a Mixed Integer Linear Programming (MILP). The model minimizes the cost of deviation from the demand for staff in each of the scenarios, while using no more than a specified number of shift types. Recall that, based on the practical case, we assume that each shift type is 1) comprised of a consecutive sequence of time periods and 2) completely contained in a given demand scenario (i.e. it does not span days). By the first assumption we will show that when the shift types used is predetermined, and thereby fixed, then we can solve the model easily by Linear Programming (LP). The second assumption allows us to decompose the problem into a master problem and a number of subproblems – one for each demand scenario – which we will exploit in Section 4.

Firstly, we assume that a set, $\mathcal{K}$, of different independent demand scenarios are given. Each scenario will typically correspond to the demand of all employees on a given day. Each scenario is divided into a set, $\mathcal{T}$, of consecutive periods. With each pair $k \in \mathcal{K}$ and $t \in \mathcal{T}$ a demand $d_{kt} \geq 0$ for staff is given. It is this demand that we have to match as closely as possible with employees working shift types in specific demand scenarios. The unit cost of undercovering a period is c^u, while the unit cost of overcovering a period is c^o. These unit costs scale linearly.

Secondly, we assume that a set, $\mathcal{S}$, of possible shift types is also given. For a period $t \in \mathcal{T}$ we denote $\mathcal{S}_t \subseteq \mathcal{S}$ the set of shift types covering the period t. Likewise we denote the set $\mathcal{T}_s \subseteq \mathcal{T}$ as the periods covered by shift type $s \in \mathcal{S}$. In the CCSDP one is allowed to use at most S^{max} such shift types to cover the demand. Similarly, we assume that the number of employees that can be used on any given day is at most E^{max}. Finally an upper bound on the number of employees on shift type $s \in \mathcal{S}$ used for a specific demand scenario $k \in \mathcal{K}$ is denoted M_{ks}. This upper bound can be set to

$$M_{ks} = \max \{ d_{kt} \,|t \in \mathcal{T}_s \},$$

i.e. the largest demand in any period that the shift type covers for the respective demand scenario.

Initially we use four types of variables. The first type of variable, $y_s \in \{0, 1\}$, is a binary variable equal to one if and only if shift type $s \in \mathcal{S}$ is available for use. We denote $\mathbf{y}$ the

vector $(y_s)_{s\in\mathcal{S}}$. The second type of variable, $x_{ks} \in \mathbb{Z}_+$, is a non-negative integer variable that counts the number of employees working shift type $s \in \mathcal{S}$ for demand scenario $k \in \mathcal{K}$. The final two types of variables are $u_{kt} \geq 0$ and $o_{kt} \geq 0$. These measure the undercover and overcover, respectively, of the demand in period $t \in \mathcal{T}$ relative to demand scenario $k \in \mathcal{K}$. The base model can now be written as:

$$\min \sum_{k\in\mathcal{K}} \left(\sum_{t\in\mathcal{T}} c^u u_{kt} + \sum_{t\in\mathcal{T}} c^o o_{kt} \right) \tag{1}$$

$$\text{s.t.} \sum_{s\in\mathcal{S}_t} x_{ks} - o_{kt} + u_{kt} = d_{kt}, \qquad k \in \mathcal{K}, t \in \mathcal{T} \tag{2}$$

$$\sum_{s\in\mathcal{S}} x_{ks} \leq E^{max}, \qquad k \in \mathcal{K} \tag{3}$$

$$x_{ks} - M_{ks} y_s \leq 0, \qquad k \in \mathcal{K}, s \in \mathcal{S} \tag{4}$$

$$\sum_{s\in\mathcal{S}} y_s \leq S^{max} \tag{5}$$

$$x_{ks} \geq 0, \qquad k \in \mathcal{K}, s \in \mathcal{S} \tag{6}$$

$$u_{kt}, o_{kt} \geq 0, \qquad k \in \mathcal{K}, t \in \mathcal{T} \tag{7}$$

$$x_{ks} \in \mathbb{Z}, \qquad k \in \mathcal{K}, s \in \mathcal{S} \tag{8}$$

$$y_s \in \{0, 1\}, \qquad s \in \mathcal{S} \tag{9}$$

The model minimizes the cost of deviating from the demand for each of the demand scenarios. This is expressed in the objective (1). The matching of the demand with number of employees assigned to shifts is given by (2), and the maximal number of employees used for a given demand scenario is enforced by (3). Constraints (4) ensure that staff can only be assigned to a shift type if the respective shift type is available for use, while Constraint (5) sets an upper bound on the number of distinct shift types we are allowed to use. We will refer to Constraint (5) as the cardinality constraint. Finally, Constraints (6)-(9) give the domains of the variables.

Model (1)-(9) is complicated by being connected across demand scenarios, i.e. by Constraints (4) and (5). If these constraints were not present, the model would decompose into $|\mathcal{K}|$ independent problems where Constraint (4) would simply become an upper bound on the x_{ks}-variables. This would actually be redundant as it would never be possible to have more employees working the shift type than the upper limit M_{ks}. On the other hand, if Constraint (4) and Constraint (5) are both included, then the y_s tend to have small fractional values in the LP-relaxation, making this LP-relaxation weak.

The upper bound on the number of employees used for any demand scenario further complicates the model. As we will argue in the following; if we did not have the connections across the scenarios, nor the upper limit on the number of employees, the problem would split into $|\mathcal{K}|$ independent easy problems to solve. Without Constraints (4) and (5), the daily demand scenarios are completely independent and could each be solved in isolation to obtain the optimal solution.

Denote $\overline{\mathbf{y}} \in \{0, 1\}^{|\mathcal{S}|}$ an arbitrary selection of shift types inducing the set $\overline{\mathcal{S}} \subseteq \mathcal{S}$ where $s \in \overline{\mathcal{S}}$ if and only if $\overline{y}_s = 1$. Define the polyhedron of all feasible allocations of employees to shift types as well as over and undercover relative to the set $\overline{\mathcal{S}}$ as

$$P_k(\overline{\mathbf{y}}) = \left\{ (\mathbf{x}_k, \mathbf{u}_k, \mathbf{o}_k) \in \mathbb{R}^{|\mathcal{S}|+2|\mathcal{T}|} \;\middle|\; \begin{array}{rll} \sum_{s \in \mathcal{S}_t} x_{ks} - o_{kt} + u_{kt} & = d_{kt}, & t \in \mathcal{T} \\ x_{ks} & \leq M_{ks}\overline{y}_s, & s \in \mathcal{S} \\ x_{ks} & \geq 0, & s \in \mathcal{S} \\ o_{kt}, u_{kt} & \geq 0, & t \in \mathcal{T} \end{array} \right\}$$

Minimizing the under and overcoverage using the constraints of $P_k(\overline{\mathbf{y}})$ results in an integer selection of the number of employees assigned to shifts as well as an integer under and overcoverage. This is summarized in the following proposition:

Proposition 1. *If all shift types $s \in \mathcal{S}$ cover a consecutive sequence of periods $t \in \mathcal{T}$ and if both d_{kt} and M_{ks} are integer then $P_k(\overline{\mathbf{y}})$ is an integer polyhedron for an arbitrary choice of $\overline{\mathbf{y}} \in \{0,1\}^{|\mathcal{S}|}$.*

Proof. As d_{kt} and $M_{ks}\overline{y}_s$ are assumed integer, it is sufficient to show that the coefficient matrix is totally unimodular. Denote $\mathbf{A} = [a_{ts}]_{t\in\mathcal{T}, s\in\mathcal{S}} = [\mathbf{a}_s]_{s\in\mathcal{S}}$ the $|\mathcal{T}| \times |\mathcal{S}|$-matrix where

$$a_{ts} = \begin{cases} 1, & s \in \mathcal{S}_t \\ 0, & \text{otherwise} \end{cases}$$

Then we can write the coefficient matrix of the polyhedron $P_k(\overline{\mathbf{y}})$ as

$$\begin{bmatrix} \mathbf{A} & -\mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{0}$ is a matrix comprised of zeros only. Any column in the matrix $\mathbf{A}$ has the structure that exactly one consecutive sequence of ones is present, i.e. for a given s the column is either $\mathbf{a}_s = [0,\ldots,0,1,\ldots,1]^t$, $\mathbf{a}_s = [1,\ldots,1,0,\ldots,0]^t$, or $\mathbf{a}_s = [0,\ldots,0,1,\ldots,1,0,\ldots,0]^t$. This is due to all shift types covering a consecutive sequence of periods. A matrix having the structure that $\mathbf{A}$ has is a so-called interval matrix. Schrijver (1986) argues that interval matrices are totally unimodular. Furthermore Schrijver (1986) observes that totally unimodularity is preserved when adding a row or a column having exactly one non-zero entry with value 1 or -1. This is exactly how the coefficient matrix of $P_k(\overline{\mathbf{y}})$ is composed, and it is therefore totally unimodular. □

A consequence of proposition 1 is that we can relax the integrality of the variables x_{ks} which count the number of employees on a given shift in the case where no upper bound on the number of employees is present, i.e. where (3) is relaxed, and still obtain an integer solution for model (1)-(9) as long as $\mathbf{y}$ is integer. This motivates us to set up a Benders decomposition in which the y-variables are treated as integer variables, while the remaining variables are considered continuous.

If undercoverage were not possible, i.e. if $u_{kt} = 0$ for all k and t, then $P_k(\overline{\mathbf{y}})$ can be formulated as a min-cost flow problem, see Segal (1974), which is totally unimodular. In our case, however, undercovering is allowed, and the min-cost flow formulation does not apply directly. Furthermore, the problem indeed has an upper bound on the number of employees allowed and adding this constraint to $P_k(\overline{\mathbf{y}})$, may impede the total unimodularity of the polytope.

4. Benders Decomposition

Benders decomposition, Benders (1962), is a method for exploiting the structure of mathematical programming problems. It specifically targets problems with a so-called *Dual Block Angular structure*. Problems having this structure break into smaller, independent problems once a set of variables is fixed. Generic descriptions of the Benders decomposition principle can be found in e.g. Geoffrion (1972) or Dirickx and Jennegren (1979). It has become a popular technique for solving certain types of complex problems, including e.g. stochastic programming problems and mixed-integer non-linear programming problems. It has also been used on problems more closely related to shift scheduling. For example, Rekik et al. (2004) solves the tour scheduling problem using Benders decomposition, and Cordeau et al. (2001) use Benders decomposition to generate schedules for flight crew as well as generate routes for aircraft.

The model for the shift design problem is composed of four types of variables – the binary y-variables, the integer x-variables and the continuous u-variables and o-variables. Suppose that we relax the integrality of the x-variables, then Benders decomposition of the problem can be applied such that only the $\mathbf{y}$-vector is considered in the master problem, while the remaining variables are considered in a set of subproblems. The master problem is used to construct solutions for the y-variables, while the subproblems are used to separate violated valid inequalities which can be used in the master problem. We will adapt this approach to the CCSDP.

Suppose that we can identify a subset of shift types such that $\overline{\mathcal{S}} \subseteq \mathcal{S}$ where $|\overline{\mathcal{S}}| \leq S^{max}$ i.e. a subset of shift types satisfying the cardinality constraint. Denote $\overline{\mathbf{y}}$ the induced solution having

$$\overline{y}_s = \begin{cases} 1, & s \in \overline{\mathcal{S}} \\ 0, & s \notin \overline{\mathcal{S}} \end{cases}$$

For $\overline{\mathbf{y}}$ fixed the problem (1)-(9) reduces to $|\mathcal{K}|$ independent problems – one for each $k \in \mathcal{K}$ and these independent problems can be stated as follows:

$$\min \sum_{t \in \mathcal{T}} c^u u_{kt} + \sum_{t \in \mathcal{T}} c^o o_{kt} \tag{10}$$

$$\text{s.t.} \sum_{s \in \mathcal{S}_t} x_{ks} - o_{kt} + u_{kt} = d_{kt}, \qquad t \in \mathcal{T} \tag{11}$$

$$\sum_{s \in \mathcal{S}} x_{ks} \leq E^{max} \tag{12}$$

$$x_{ks} \leq M_{ks} \overline{y}_s, \quad s \in \mathcal{S} \tag{13}$$

$$x_{ks} \geq 0, \qquad s \in \mathcal{S} \tag{14}$$

$$u_{kt}, o_{kt} \geq 0, \qquad t \in \mathcal{T} \tag{15}$$

$$x_{ks} \in \mathbb{Z}, \qquad s \in \mathcal{S} \tag{16}$$

Model (10)-(16) identifies the best allocation of employees to shifts for each scenario $k \in \mathcal{K}$ when the subset of allowable shifts is given by $\overline{\mathcal{S}}$. It can be observed that this model always has a feasible solution e.g. by fixing $x_{ks} = 0$ for all $s \in \mathcal{S}$ and put $u_{kt} = d_{kt}$ for all periods $t \in \mathcal{T}$. Furthermore, the model will have a bounded optimum as long as $c^o, c^u \geq 0$ (even though unbounded alternative solutions may exist when either $c^o = 0$ or $c^u = 0$). This is also true for the LP-relaxation i.e. if we relax constraint (16), the model still has at least one bounded optimal solution.

We apply Benders decomposition on the LP-relaxation of problem (10)-(16). The motivation for this is that if we further relax Constraint (12) then the resulting problem will always yield an integer solution, as shown in Proposition 1. This is also true in the case where (12) is not binding. In the following we will denote the problem (10)-(15) for the primal subproblem.

We define $\pi_{kt} \in \mathbb{R}$, $\mu_k \leq 0$, and $\gamma_{ks} \leq 0$ to be the dual variables of constraints (11), (12), and (13), respectively. Then the dual of the primal subproblem of (10)-(15) becomes

$$\max \quad \sum_{t \in \mathcal{T}} d_{kt}\pi_{kt} + E^{max}\mu_k + \sum_{s \in \mathcal{S}} M_{ks}\overline{y}_s\gamma_{ks} \tag{17}$$

$$\text{s.t.} \quad \sum_{t \in \mathcal{T}_s} \pi_{kt} + \mu_k + \gamma_{ks} \leq 0, \qquad s \in \mathcal{S} \tag{18}$$

$$-c^o \leq \pi_{kt} \leq c^u, \qquad t \in \mathcal{T} \tag{19}$$

$$\mu_k \leq 0, \tag{20}$$

$$\gamma_{ks} \leq 0, \qquad s \in \mathcal{S} \tag{21}$$

and we denote this problem the dual subproblem for scenario $k \in \mathcal{K}$. We can observe that only the objective of the dual problem changes when changing demand scenario or the allowable shift set. Thus, the polyhedron of feasible dual solutions for each of the primal subproblems (10)-(15) is identical, and we denote this polyhedron:

$$\mathcal{D} = \left\{ (\boldsymbol{\pi}, \mu, \boldsymbol{\gamma}) \in \mathbb{R}^{|\mathcal{T}|+1+|\mathcal{S}|} \;\middle|\; \begin{array}{ll} \sum_{t \in \mathcal{T}_s} \pi_t + \mu + \gamma_s \leq 0, & s \in \mathcal{S} \\ -c^o \leq \pi_t \leq c^u, & t \in \mathcal{T} \\ \mu \leq 0, & \\ \gamma_s \leq 0, & s \in \mathcal{S} \end{array} \right\}$$

where $\boldsymbol{\pi} = (\pi_t)_{t \in \mathcal{T}}$ and $\boldsymbol{\gamma} = (\gamma_s)_{s \in \mathcal{S}}$. We observe that $(\mathbf{0}, 0, \mathbf{0}) \in \mathcal{D}$ and consequently $\mathcal{D} \neq \emptyset$. The dual polyhedron is unbounded. However, if the objective coefficients of the dual objective (17) are non-negative, then the dual problem (17)-(21) will have a bounded optimum.

We denote the index set of extreme points of $\mathcal{D}$ as $\mathcal{P}$ and we write the extreme points as $(\boldsymbol{\pi}^p, \mu^p, \boldsymbol{\gamma}^p) \in \mathcal{D}$ for $p \in \mathcal{P}$. Each dual extreme point yields a so-called Benders (or optimality) cut for each of the demand scenarios $k \in \mathcal{K}$

$$z_k \geq \sum_{t \in \mathcal{T}} d_{kt}\pi_t^p + E^{max}\mu^p + \sum_{s \in \mathcal{S}} M_{ks}y_s\gamma_s^p, \quad p \in \mathcal{P}, k \in \mathcal{K} \tag{22}$$

where z_k is a variable measuring the contribution of scenario $k \in \mathcal{K}$ to the objective of the full problem. Intuitively, the Benders cuts collectively approximate the objective of each of the demand scenarios as a convex piece-wise linear function which is referred to as an outer approximation. If a Benders cut is binding then a change in a y_k variable by Δ will result in a potential change of the objective of $\Delta M_{ks}\gamma_s^p$. Thus, the Benders cuts can be used to indicate which shift types, through the y-variables, are good to have in a solution.

The Benders reformulation of the problem (1)-(9) with integrality of the x-variables

relaxed is then

$$\min \sum_{k \in \mathcal{K}} z_k \tag{23}$$

$$\text{s.t.} \sum_{s \in \mathcal{S}} y_s \leq S^{max} \tag{24}$$

$$z_k - \sum_{s \in \mathcal{S}} M_{ks} \gamma_s^p y_s \geq \sum_{t \in \mathcal{T}} d_{kt} \pi_t^p + E^{max} \mu^p, \quad p \in \mathcal{P}, k \in \mathcal{K} \tag{25}$$

$$y_s \in \{0, 1\}, \qquad s \in \mathcal{S} \tag{26}$$

The objective, (23), measures the contribution to the cost of each demand scenario $k \in \mathcal{K}$ for the given selection of allowable shift types. The first constraint, (24), states that no more than S^{max} shift types are allowed and it is equivalent with Constraint (5) of the base model. The full set of Benders cuts are given in constraint (25) which is a reordered version of Constraint (22) where all constants appear on the right-hand side. Finally, the Benders reformulation maintains the binary nature of the shift type variables y_s in constraint (26). The number of Benders cuts is typically exponential in size compared to the number of constraints in the dual polyhedron $\mathcal{D}$, and computationally we only use a small subset of all of the Benders cuts. We let the index set $\mathcal{B} \subseteq \mathcal{P} \times \mathcal{K}$ correspond to the Benders cuts we use at any given time. Thus we relax constraints (25), only including cuts with indices in $\mathcal{B}$.

In practice we separate the Benders cuts (22) by solving the primal problem (10)-(15), as we can then directly verify whether or not the Benders cut corresponds to a subproblem solution having integer x-values, in which case the corresponding solution will be feasible for the original problem.

4.1. Dual alternative solutions

The subproblems (10)-(15) are severely degenerate. This is due to the cardinality constraint, which typically only allows a small number of y-variables to be non-zero in an optimal (integer) solution. As a consequence of the degeneracy the dual optimal solution is rarely unique; i.e. a set of optimal dual alternative solutions typically exists. The Benders cuts (22) rely on these dual optimal solutions and multiple Benders cuts can be generated in this case.

Magnanti and Wong (1981) observe that it is important to generate good Benders cuts which are not dominated by other Benders cuts. As the Benders cuts depend on the dual optimal solution we generate several diverse dual alternative solutions. We use an approach inspired by Rousseau et al. (2007), to obtain these dual alternative solutions.[1] First we solve the primal version of the subproblem, and observe that we can obtain the polyhedron of dual alternative optimal solutions using the complementary slackness conditions. From this polyhedron we select several dual solutions.

Suppose that we have a solution $(\mathbf{x}_k^*, \mathbf{u}_k^*, \mathbf{o}_k^*)$ to the subproblem (10)-(15) for a given shift type solution $\overline{\mathbf{y}}$ and a given scenario $k \in \mathcal{K}$. The complementary slackness conditions impose additional constraints on the dual optimal solution given by the following polytope:

[1] Rousseau et al. (2007) identifies dual alternative solutions in order to find interior points of the dual optimal polyhedron, and then they use these interior points for stabilization in column generation.

$$C_k(\mathbf{x}_k^*, \mathbf{u}_k^*, \mathbf{o}_k^*, \overline{\mathbf{y}}) = \left\{ (\boldsymbol{\pi}, \mu, \boldsymbol{\gamma}) \in \mathbb{R}^{|\mathcal{T}|+1+|\mathcal{S}|} \;\middle|\; \begin{array}{ll} \sum_{t \in \mathcal{T}_s} \pi_t + \mu + \gamma_s = 0, & s \in \mathcal{S} : x_{ks}^* > 0 \\ \pi_t = -c^o, & t \in \mathcal{T} : o_{kt}^* > 0 \\ \pi_t = c^u, & t \in \mathcal{T} : u_{kt}^* > 0 \\ \mu = 0, & \text{if } \sum_{s \in \mathcal{S}} x_{ks}^* < E^{max} \\ \gamma_s = 0, & s \in \mathcal{S} : x_{ks} < M_{ks} \overline{y}_s \end{array} \right\}$$

Hence, given a $\overline{\mathbf{y}}$ and a primal optimal (and possibly degenerate) solution $(\mathbf{x}_k^*, \mathbf{u}_k^*, \mathbf{o}_k^*)$, then the set of dual optimal alternative solutions is given by

$$\mathcal{D}_k^*(\mathbf{x}_k^*, \mathbf{u}_k^*, \mathbf{o}_k^*, \overline{\mathbf{y}}) = \mathcal{D} \cap C_k(\mathbf{x}_k^*, \mathbf{u}_k^*, \mathbf{o}_k^*, \overline{\mathbf{y}})$$

and each point within this polyhedron will correspond to a Benders cut having equally large absolute violation of the objective. It is from this set we are interested in obtaining a diverse set of points. As all points within $\mathcal{D}_k^*(\mathbf{x}_k^*, \mathbf{u}_k^*, \mathbf{o}_k^*, \overline{\mathbf{y}})$ yield the same absolute violation we can search for points $(\boldsymbol{\pi}, \mu, \boldsymbol{\gamma}) \in \mathcal{D}_k^*(\mathbf{x}_k^*, \mathbf{u}_k^*, \mathbf{o}_k^*, \overline{\mathbf{y}})$ having different characteristics. Let $\mathbf{w}^1 \in \mathbb{R}^{|\mathcal{T}|}$, $w^2 \in \mathbb{R}$, and $\mathbf{w}^3 \in \mathbb{R}^{|\mathcal{S}|}$ be weights assigned to $\boldsymbol{\pi}$, μ ,and $\boldsymbol{\gamma}$ respectively. Then we solve the following problem

$$\max \quad \mathbf{w}^1 \boldsymbol{\pi} + w^2 \mu + \mathbf{w}^3 \boldsymbol{\gamma} \tag{27}$$

$$\text{s.t.} \quad (\boldsymbol{\pi}, \mu, \boldsymbol{\gamma}) \in \mathcal{D}_k^*(\mathbf{x}_k^*, \mathbf{u}_k^*, \mathbf{o}_k^*, \overline{\mathbf{y}}) \tag{28}$$

The weights $\mathbf{w}^1$ of the demand duals $\boldsymbol{\pi}$ can take any real value, whereas the weights of the w^2 and $\mathbf{w}^3$ have to be non-negative in order to obtain bounded solutions.

To obtain several dual alternative solutions we solve (27)-(28) for different settings of the weights $\mathbf{w}^1$, w^2, and $\mathbf{w}^3$. We initialize $\mathbf{w}^1 = (-1, \ldots, -1)$, $w^2 = 0$ and $\mathbf{w}^3 = (1, \ldots, 1)$. Selecting a negative value will tend to minimize the corresponding dual, while selecting a positive value will tend to maximize the corresponding dual. Leaving the weight at zero indicates that we have no preferences on the size or sign of the corresponding dual. Our selection hence tries to minimize $\boldsymbol{\pi}$ and maximize $\boldsymbol{\gamma}$. The shift duals, $\boldsymbol{\gamma}$, are upper bounded by zero. If $\gamma_s = 0$, this value is attained, and implies that $M_{ks} y_s \gamma_s = 0$. In such cases there is no contribution to the corresponding Benders cut (22), regardless of the value y_s.

When having found a dual solution $(\boldsymbol{\pi}', \mu', \boldsymbol{\gamma}')$ by (27)-(28) then the cut, if not already been found, is added to $\mathcal{B}$ and the weights are adjusted. This adjustment follows the simple rule that if $\gamma_s' = 0$ and $w_s^3 = 1$ then we put $w_s^3 = 0$. The process is then repeated by solving (27)-(28) obtaining Benders cuts and adjusting the weights. It is terminated either when $w_s^3 = 0$ for all $\gamma_s' = 0$ and thereby no adjustment is conducted, or when a satisfactory number of dual alternative solutions has been obtained. The adjustment of the weights results in different sets of shift duals being zero and consequently different y-variables will have an effect on the z_k-variable values.

5. Benders based heuristic

Despite the structure of Model (1)-(9) lending itself very naturally to Benders decomposition, a conventional implementation of this is unlikely to perform well for the CCSDP. There are two possibilities when implementing Benders decomposition, either an iterative approach or a Branch and Cut (BAC) based approach. The first solves the master problem

and subproblems iteratively, and usually requires that the master problem is solved to integer optimality at every iteration. The time to solve the master problem naturally increases as the number of Benders cuts in the formulation of Model (23)-(26) increases. A BAC approach, on the other hand, treats the subproblems as a separation routine for the master problem and, in addition to integer solutions, also identifies violated Benders cuts at relaxed solutions, embedding the entire process in a Branch and Bound (BAB) tree. In comparison, the BAC approach can be thought of as solving one MILP, compared to the iterative approach, which solves a sequence of MILPs. Neither approach is particularly appealing here. The large number of shift type variables in the master problem suggests a sequence of potentially slow MILPs must be solved for the iterative approach, while the fractional possibilities and lack of costs on the y-variables indicate that a large BAB tree may be encountered with the BAC approach. Furthermore, as already mentioned in Section 4.1, in both cases there are likely to be many dual alternative optimal solutions for a given solution to the master problem, each of which can be used to construct a Benders cut. Combining this large set of potentially weak Benders cuts with the fact that up to $|\mathcal{K}|$ subproblems need to be solved at every iteration, means that that convergence of the Benders decomposition will be slow. Preliminary experiments with an iterative approach confirmed these observations. For extremely large problems, however, Benders decomposition does make it possible to find feasible solutions (of a proven quality), while a direct MILP formulation cannot.

Given the expected poor performance of Benders decomposition on a large CCSDP, as well as the fact that S^{max} is typically much smaller than $|\mathcal{S}|$, we propose an iterative heuristic framework based on Benders decomposition which at any given time considers only a very small set of shift types. This set is then dynamically updated when we detect a shift type not currently considered, but with the potential to improve the solution quality. This detection is based on the cuts generated when solving the Benders decomposition for the small subset of shift types. In what follows we describe the heuristic framework in detail; an outline of the framework is sketched in Algorithm 1.

The approach begins with the construction of the initial set of shifts types. This is a heuristic procedure that selects S^{max} shift types and tries to identify a "good" set of shift types. Shift types are assigned a score and selected iteratively. The score is based on several characteristics of the shift types: (1) the number of uncovered periods the shift type covers, ensuring all periods get covered by at least one shift type; (2) the increase in demand (aggregated over scenarios) of the upcoming periods after the shift type has started. When the demand increases significantly in some periods, more staff are needed and therefore it is reasonable to start a shift type; (3) the decrease in demand (aggregated over scenarios) of the upcoming periods after ending the shift type. When the demand is decreasing then it should be possible to decrease the number of employees and therefore a shift type should end close to a large decrease in demand; (4) the length of the shift type (longer is better). The longer a shift type is the more demand it may cover and therefore these are initially preferred; (5) the number of periods from the start of the shift type to the start of an already inserted shift type (more is better); and (6) the number of periods from the end of the shift type to the end of an already inserted shift type (more is better). The last two characteristics are used to get an even spread of starting times and ending times. The heuristic selects the shift type with the largest score and inserts it into the initial set of shift types and the scores for the remaining shift types are adjusted. This is repeated until S^{max} shift types are selected.

The premise is that one can quickly determine, via Benders decomposition, the optimal solution to the problem considering a small set of shift types $\hat{\mathcal{S}}$, where $|\hat{\mathcal{S}}|$ is only slightly

Algorithm 1 The Benders Decomposition Based Heuristic

procedure BENDERSHEURISTIC($\mathcal{S}$, $\mathcal{K}$)
  $\hat{\mathcal{S}} \leftarrow$ *initialShiftSet()*
  $\hat{z}_{prev} \leftarrow \infty$, $\mathcal{S}_{pot} \leftarrow \emptyset$, $\mathcal{S}^{+}_{prev} \leftarrow \emptyset$
  solved $\leftarrow$ `false`
  while `time exists` and `not` solved **do**
    $\mathcal{S}^{+} \leftarrow \emptyset$
    $(\hat{z}, \hat{\mathbf{y}}, \hat{\mathcal{B}}) \leftarrow$ *BendersDecomposition*$(\hat{\mathcal{S}})$
    if $\hat{z} < \hat{z}_{prev}$ **then**
      $(\mathcal{S}_{pot}, \boldsymbol{\delta}^{\min}) \leftarrow$ *getNewShifts*$(\hat{\mathcal{B}}, \hat{\mathbf{y}}, \hat{\mathcal{S}}, \mathcal{S}, \mathcal{K})$
      $\hat{\mathcal{S}} \leftarrow \hat{\mathcal{S}} \setminus \{s | \hat{y}_s < \epsilon\}$
    else
      $\hat{\mathcal{S}} \leftarrow \hat{\mathcal{S}} \setminus \mathcal{S}^{+}_{prev}$
    if $\mathcal{S}_{pot} \neq \emptyset$ **then**
      while $\mathcal{S}_{pot} \neq \emptyset$ `and` $i \leq maxShifts$ **do**
        $s_{pot} = \text{argmin}_{s \in \mathcal{S}_{pot}} \{\delta^{\min}_s\}$
        $\mathcal{S}^{+} \leftarrow \mathcal{S}^{+} \cup \{s_{pot}\}$
        $\mathcal{S}_{pot} \leftarrow \mathcal{S}_{pot} \setminus \{s_{pot}\}$
        $i \leftarrow i + 1$
      $\hat{\mathcal{S}} \leftarrow \hat{\mathcal{S}} \cup \mathcal{S}^{+}$
    else
      solved $\leftarrow$ `true`
    $\mathcal{S}^{+}_{prev} \leftarrow \mathcal{S}^{+}$
    $\hat{z}_{prev} \leftarrow \hat{z}$
  return $\hat{\mathbf{y}}$

larger than S^{max}, and that the separated Benders cuts can be used to identify shift types that should be included in $\hat{\mathcal{S}}$. That is, we fix $y_s = 0$ for all $s \in \mathcal{S} \setminus \hat{\mathcal{S}}$ and then solve the resulting restricted problem. This is done by a conventional Benders algorithm where in each Benders iteration the master problem (23)-(26) is solved to integer optimality after which the subproblems (10)-(15) for each demand scenario are solved sequentially to obtain violated Benders cuts. This process continues until no more violated Benders cuts can be found for the integer solution, and the integer solution $\hat{\mathbf{y}}$ is then optimal for the problem restricted to $\hat{\mathcal{S}}$. In this way we can restrict attention to promising shift types only, i.e. those shift types in $\hat{\mathcal{S}}$, and obtain a solution $(\hat{\mathbf{z}}, \hat{\mathbf{y}})$ where $\hat{y}_s = 0$ for all $s \in \mathcal{S} \setminus \hat{\mathcal{S}}$ and $\hat{y}_s \in \{0, 1\}$ for $s \in \hat{\mathcal{S}}$, and where $\hat{\mathbf{z}} = (\hat{z}_k)_{k \in \mathcal{K}}$ is the vector of contributions to the objective for each demand scenario. We will denote $\hat{z} = \sum_{k \in \mathcal{K}} \hat{z}_k$ i.e. $\hat{z}$ is the objective value of the solution. In addition to the solution we also obtain a set of Benders cuts $\hat{\mathcal{B}}$.

As the solution $\hat{\mathbf{y}}$ satisfies the cardinality constraint and as $|\hat{\mathcal{S}}|$ is slightly larger than S^{max} we must have that some $\hat{y}_s = 0$ for $s \in \hat{\mathcal{S}}$. If the objective $\hat{z}$ is better than the previously found objective, these shifts are removed from the set of promising shifts $s \in \hat{\mathcal{S}}$. In this way, we can ensure that the size of $\hat{\mathcal{S}}$ remains manageable. The resulting set of Benders cuts $\hat{\mathcal{B}}$ reflects the outer approximation of the objective locally around the solution $\hat{\mathbf{y}}$, and it can be used to approximate the change in the objective when a small change to the solution is made. In the flowing we describe how we use the set $\hat{\mathcal{B}}$ to obtain new promising shift types to include in $\hat{\mathcal{S}}$.

Given an optimal solution $\hat{\mathbf{y}}$ to the reduced problem, the next step of the heuristic involves identifying a set of promising shift types $\mathcal{S}^{+}$ to add to $\hat{\mathcal{S}}$ and their potential im-

provements. Algorithm 2 describes in detail how this is done. The procedure analyses the set of Benders cuts just found. It begins by iterating over all binding cuts, since these collectively determine the solution quality; a binding Benders cut for a given scenario k determines the value of $\hat{z}_k$. For each binding cut and pair of shifts ($s_1 \in \hat{\mathcal{S}}, s_2 \in \mathcal{S} \setminus \hat{\mathcal{S}}$) we determine an estimate for the improvement (or deterioration)

$$\delta = M_{ks_2}\gamma^p_{s_2} - M_{ks_1}\gamma^p_{s_1}$$

in z_k by swapping s_2 with s_1 in the solution. Note that even though we have only included a very small subset of shift types in the subproblems the Benders cuts generated include information, $\gamma^p_{s_2}$, on non-included shift type variables. Therefore, since the Benders cuts are based on dual information, the value $M_{ks}\gamma^p_s$ for any shift variable y_s can be used to determine the approximate change in objective by forcing y_s into the solution. As multiple Benders cuts may be binding for each of the scenarios we record the maximum change, $\Delta_{s_1s_2k}$, for the respective scenario. This is due to the fact that only the maximum value of the right-hand sides of the Benders cuts (22) will determine z_k.

Algorithm 2 Identifying Promising Shifts

```
1:  procedure getNewShifts(B, ŷ, Ŝ, S, K)
2:      Δ ← {−∞}, δ^min ← {∞}
3:      S_pot ← ∅
4:      for (p, k) in B do
5:          if z_k = Σ_{t∈T} d_kt π^p_t + E^max μ^p + Σ_{s∈S} M_ks ŷ_s γ^p_s then
6:              for s_1 in Ŝ do
7:                  for s_2 in S \ Ŝ and ŷ_{s_1} > ε do
8:                      δ ← M_{ks_2} γ^p_{s_2} − M_{ks_1} γ^p_{s_1}
9:                      Δ_{s_1 s_2 k} ← max(Δ_{s_1 s_2 k}, δ)
10:     for s_2 in S \ Ŝ do
11:         for s_1 in Ŝ do
12:             δ ← 0
13:             for k in K do
14:                 if Δ_{s_1 s_2 k} > −∞ then
15:                     δ ← δ + Δ_{s_1 s_2 k}
16:             if δ < δ^min_{s_2} then
17:                 δ^min_{s_2} ← δ
18:         if δ^min_{s_2} < −ε then
19:             S_pot ← S_pot ∪ {s_2}
    return (S_pot, δ^min)
```

Once all such scenario objective changes have been computed, we loop over all shift types $s_2 \in \mathcal{S} \setminus \hat{\mathcal{S}}$ and identify whether or not including s_2 instead of $s_1 \in \hat{\mathcal{S}}$ in the solution would produce a negative net change across all demand scenarios. It could be the case that a shift type is very attractive for one scenario, but not so attractive for other scenarios. If the estimated net objective change is negative, however, we would still like to tag this shift type as promising. All shift types with estimated potential improvements are stored in the set $\mathcal{S}_{pot}$ and potential improvements are stored in the vector $\boldsymbol{\delta}^{\min} = (\delta^{min}_s)_{s\in\mathcal{S}}$. Note that both $\mathcal{S}_{pot}$ and $\boldsymbol{\delta}^{\min}$ are only determined if, on successive iterations, an improvement in objective value of the restricted problem is detected (see lines 8-9 of Algorithm 1). Here $\hat{z}_{prev}$ states the objective value of the previous iteration. When this occurs, any shift types not selected

in the current solution are also removed. If the objective value remains the same, then any shift types that were added on the previous iteration, denoted $\mathcal{S}^{+}_{prev}$, are removed since their inclusion did not result in an improvement. In this case the set of potential entering shift types computed on the previous iteration remains valid. Keeping $\mathcal{S}^{+}_{prev}$ makes it possible to restore the state of the previous iteration and try the next best set of potential shift types if the current $\hat{\mathcal{S}}$ does not contain a better solution.

It is important to note that there is no guarantee a shift type identified as promising actually improves the solution quality. Algorithm 2 is based on limited information only, and the shift types it identifies are, at best, only an estimate. For this reason, we do not consider removing shift types until the Benders decomposition step of Algorithm 1 has been performed; we prefer to wait and see if the newly added shift types actually yield an improvement. As a consequence, during its execution, Algorithm 1 will never produce a worse objective on a subsequent iteration. When updating $\hat{\mathcal{S}}$ with new shift types (lines 13-21, Algorithm 1), we simply choose the $maxShifts$ (or $|\mathcal{S}_{pot}|$, whichever is the smaller) most promising potential shift types of $\mathcal{S}_{pot}$. Note the $maxShifts$ is a parameter specified in advance. In practice, we set this to one; however, in general one should be able to consider any number of shift types to dynamically add.

Algorithm 1 terminates when a specified run time has elapsed or when no promising shift types are identified. As such, it converges to a locally optimal solution; however, by identifying the shift types to include using the Benders cuts, this should provide a reasonable solution.

6. Computational experiments

In this section we examine the performance of the proposed methodology on a set of realistic test instances. These instances are based on data which has been provided by Copenhagen Airport and focus on a typical week (i.e. seven different demand scenarios). Each demand scenario specifies for a given day the required staffing level in the main security hall at 15 minute intervals. This gives a total of 96 time periods per day for which there is an associated staff demand. Using this demand data we construct several instances of the CCSDP by varying the maximum number of shift types that can be used, S^{max}, and by considering different sets of shift types, $\mathcal{S}$. We also analyse the impact of varying the number of staff available, E^{max}. For each data set we compare the results obtained using four different versions of the Benders heuristic; however, the only difference between the approaches is in the number of dual alternative optimal solutions that one is allowed to find at a given iteration of the Benders decomposition component of the algorithm. By varying the number of such solutions, and by extension the number of Benders cuts returned, we assess the potential benefit in searching for dual alternative optimal solutions. Finally, to provide a performance comparison all test instances are also solved with the commercial solver Gurobi version 6.0 (single thread). All tests have been performed on Intel(R) Xeon(R) CPU X5550 @ 2.67GHz with 24GB ram running Ubuntu Linux 14.04.

Before closely examining the results, we provide more specific details on the different data sets. We consider two different sets of shift types. The first contains all possible shift types that have a duration of between four and eight hours. This gives a total of 1241 different shift types. The second set, albeit slightly unrealistic, considers all possible shift types of between 4 and 12 hours in duration. This second set contains 2145 different shift types and is included in an attempt to stress test the algorithm. From a staffing level perspective, we consider instances having 160, 180, and 200 available employees. This gives a diverse set of

instances in which it is impossible to cover all demand (but is almost possible), regardless of how many shift types are permitted. More undercoverage is expected on instances with a fewer number of employees. The staffing levels are close to what the airport has in practice.

For the three different values for E^{max} and the two different sets of allowable shift types, we create 16 different CCSDPs by varying the value of S^{max} from six to 21. This gives a total of $3 \times 2 \times 16 = 96$ different CCSDPs. Note that the demand scenarios for all data sets are the same. We solve each of these data sets five times, once with Gurobi and four times with the heuristic. Furthermore, we compare the quality of the solutions obtained when maximum runtimes of 10 minutes, 20 minutes, and one hour are enforced. In all of the tests it is assumed that a unit of undercoverage, c^u, is ten times as expensive as a unit of overcoverage, c^o. For reference, and comparative purposes, considering a relaxed version of the problem in which no upper bound is enforced on the number of different shift types allowed yields optimal solutions having 163, 173, and 173 different shift types for the three different staffing levels on instances with 1241 shift types. For the larger instances these are 171, 169, and 173, respectively. Such solutions state the number of different shift types with at least one staff member assigned on any of the given days.

Tables 1 and 2 summarize the main results. Tables 1 reports the results when $|\mathcal{S}| = 1241$, while Table 2 gives the results when $|\mathcal{S}| = 2145$. Each row of the table corresponds to a different set of instances. An instance set is characterized by a staffing level, a number of possible shift types, and a time limit. For example, instance 180-1241-600 considers 180 employees, 1241 shift types, where a run time of 600 seconds is permitted. Recall that each instance set contains 16 different CCSDPs. The tables report for each instance set, the number of times each approach found the best solution within the time limit. The "GRB" column refers to the approach where Gurobi is used to directly solve Model (1)-(7), and (9). This ensures a fair comparison between the approaches. Column "c_j" refers to the heuristic approach in which at most j additional dual alternative optimal solutions are considered at each Benders iteration. In addition, Tables 1 and 2 give the average, percentage difference between the best value found by each heuristic approach and that found by Gurobi within the allowed time limit. Given in parentheses are the number of times a heuristic outperformed Gurobi on the 16 data sets. Tables 3 and 4 provides the same results where the comparison includes a Gurobi solve of Model (1)-(9), i.e. integral x-variables.

Table 1: Results Summary - Gurobi is given Model (1)-(7), and (9)

		Found Best Value				% Change			
Instance Set	GRB	c_0	c_1	c_3	c_5	c_0	c_1	c_3	c_5
160-1241-600	1	3 (15)	4 (15)	8 (15)	7 (15)	-32.88	-34.61	-36.14	-36.20
180-1241-600	0	3 (15)	1 (16)	11 (16)	5 (16)	-40.17	-38.31	-45.59	-44.38
200-1241-600	0	3 (16)	2 (16)	10 (16)	5 (16)	-40.15	-40.02	-45.11	-45.00
160-1241-1200	0	3 (16)	7 (16)	6 (16)	5 (16)	-26.53	-27.73	-27.72	-28.28
180-1241-1200	0	4 (15)	3 (15)	8 (16)	4 (16)	-29.99	-28.50	-34.07	-32.14
200-1241-1200	1	2 (13)	0 (14)	7 (15)	8 (15)	-24.20	-23.74	-29.18	-30.15
160-1241-3600	0	3 (8)	6 (12)	5 (12)	4 (12)	-2.98	-4.66	-3.62	-6.02
180-1241-3600	0	5 (14)	1 (14)	5 (16)	6 (14)	-14.58	-14.02	-18.22	-18.31
200-1241-3600	0	3 (15)	2 (13)	8 (15)	8 (16)	-11.98	-13.42	-17.11	-16.86

From Table 1, it is clear to see that the heuristic approach is superior to that of a

Table 2: Results Summary - Gurobi is given Model (1)-(7), and (9)

		Found Best Value				% Change			
Instance Set	GRB	c_0	c_1	c_3	c_5	c_0	c_1	c_3	c_5
160-2145-600	3	3 (10)	0 (9)	6 (12)	5 (12)	-13.81	-10.61	-18.88	-18.20
180-2145-600	4	2 (11)	0 (10)	3 (12)	8 (12)	-12.26	-10.60	-16.21	-17.77
200-2145-600	2	2 (11)	2 (11)	7 (12)	4 (13)	-11.73	-13.79	-18.89	-17.34
160-2145-1200	5	3 (10)	0 (9)	4 (11)	4 (10)	-7.70	-6.50	-14.54	-13.54
180-2145-1200	4	4 (11)	0 (11)	3 (12)	6 (12)	-10.31	-9.29	-13.03	-14.91
200-2145-1200	2	2 (12)	3 (11)	5 (12)	5 (13)	-7.37	-10.33	-14.47	-13.12
160-2145-3600	6	1 (9)	1 (10)	4 (10)	4 (10)	-4.77	-4.94	-11.34	-10.29
180-2145-3600	4	3 (9)	1 (10)	5 (11)	4 (11)	-4.29	-2.36	-8.03	-7.58
200-2145-3600	2	1 (10)	3 (10)	7 (12)	3 (11)	-2.43	-5.99	-10.13	-8.25

Table 3: Results Summary - Gurobi is given Model (1)-(9)

		Found Best Value				% Change			
Instance Set	GRB	c_0	c_1	c_3	c_5	c_0	c_1	c_3	c_5
160-1241-600	0	4 (16)	4 (15)	8 (15)	7 (15)	-33.32	-34.34	-34.97	-35.23
180-1241-600	0	3 (16)	1 (16)	11 (16)	5 (16)	-37.09	-35.43	-41.02	-39.76
200-1241-600	0	3 (16)	2 (16)	10 (16)	5 (16)	-36.24	-36.18	-40.00	-40.23
160-1241-1200	2	2 (14)	6 (13)	5 (13)	3 (13)	-27.21	-27.81	-27.76	-28.29
180-1241-1200	0	4 (16)	3 (15)	8 (15)	4 (15)	-36.83	-35.34	-39.55	-37.80
200-1241-1200	1	2 (15)	0 (15)	7 (15)	8 (15)	-33.47	-33.43	-36.91	-37.90
160-1241-3600	2	2 (13)	5 (13)	5 (13)	4 (13)	-13.17	-14.56	-14.03	-16.08
180-1241-3600	0	5 (16)	1 (15)	5 (16)	6 (16)	-29.30	-28.20	-31.40	-31.47
200-1241-3600	0	3 (13)	2 (15)	8 (14)	8 (16)	-23.59	-24.64	-27.49	-27.37

Gurobi solve on instance sets containing 1241 shift types. The difference in solution values obtained on such instances is particularly pronounced when a time limit of 600 seconds is enforced; on average, all heuristic approaches yield solutions that are around 40% better than that of a direct MILP solve. Furthermore, the results suggest that there is some benefit in searching for dual alternative optimal solutions. The heuristic approaches in which more dual alternative optimal solutions (i.e. c_3 and c_5) tend to provide the best results. Interestingly, such approaches consider a fewer total number of solutions to CCDSP; however, by including more Benders cuts at each iteration, more reliable information is available when determining new shift types to consider. The longer we let the algorithms run the smaller the percentage difference between the best found heuristic solutions and the best found MILP solutions. With the exception of the instances containing 160 employees perhaps, all heuristic solutions are around 12-18% better after an hour of computing time, and in the majority of cases have found a better solution than the MILP solve. Even for the instances with 160 employees, all heuristics return a better solution in at least half of the cases; however, the improvement is not as pronounced, sitting at around 3-6%. The direct MILP solve does return the best solution on two rare occasions.

Looking at Table 2, the comparison is not so one-sided; however, the results still suggest

Table 4: Results Summary - Gurobi is given Model (1)-(9)

		Found Best Value				% Change			
Instance Set	GRB	c_0	c_1	c_3	c_5	c_0	c_1	c_3	c_5
160-2145-600	0	4 (16)	0 (16)	7 (16)	7 (16)	-40.97	-39.01	-44.76	-44.38
180-2145-600	0	2 (16)	1 (16)	6 (16)	10 (16)	-40.02	-39.14	-42.61	-44.02
200-2145-600	0	2 (15)	2 (16)	9 (16)	6 (16)	-40.13	-41.52	-45.04	-44.03
160-2145-1200	0	4 (15)	0 (15)	6 (15)	6 (15)	-37.47	-36.97	-42.37	-41.69
180-2145-1200	0	4 (16)	1 (16)	6 (16)	7 (16)	-36.32	-36.06	-38.42	-40.04
200-2145-1200	0	2 (15)	3 (16)	6 (16)	6 (16)	-37.75	-39.71	-42.39	-41.64
160-2145-3600	0	2 (15)	1 (15)	8 (16)	6 (16)	-32.21	-32.31	-36.99	-36.09
180-2145-3600	0	4 (16)	1 (16)	8 (16)	5 (16)	-35.43	-34.08	-37.67	-37.69
200-2145-3600	0	1 (16)	5 (16)	8 (16)	3 (16)	-34.63	-37.32	-39.51	-38.61

that it is beneficial to consider dual alternative optimal solutions if running the heuristic. A direct MILP solve using Gurobi is much more competitive on these instance sets, finding better solutions than the heuristic on more instances (compared to the instances containing 1241 shift types). That said, all heuristic approaches still outperform Gurobi by 3-11% on average after an hour of computing time. A possible explanation for the closer comparison could be that with 2145 shift types the Benders decomposition subproblems are larger. Larger subproblems take longer to solve, and this in turn slows down the Benders decomposition step of the heuristic. As a consequence, fewer solutions to the CCSDP can be considered for such instances within the allotted time. Given the decreasing performance improvement of the heuristics over a direct MILP solve, a natural question to ask is how far from optimality the final solutions are. The heuristic approaches do not provide any such information; however, optimality gaps are available for the MILP solves. These are reported in Table 5. This table reports for both model types and combinations of staffing level and shift types all average optimality gaps for the instance sets.

Table 5: Average Optimality Gaps in percent at 3600s

	$\lvert\mathcal{S}\rvert = 1241$			$\lvert\mathcal{S}\rvert = 2145$		
Model	160	180	200	160	180	200
(1)-(7), and (9)	65.31	81.52	76.59	85.99	85.28	85.50
(1)-(9)	84.80	92.57	91.35	93.35	93.59	93.74

Table 5 shows that, even after 1-hour, there are significant optimality gaps that must be closed to prove optimality. Given such high percentages, it seems very unlikely that optimality will be proved quickly. The values in Table 5 highlight the weakness of the MILP formulation. Naturally, the more time we give the algorithms, the closer the percentage improvements will be. Obviously, if it were possible to solve such problems to optimality the direct MILP solve would eventually get there, while the same is not necessarily true of the heuristic approaches. It is, however, encouraging to see that the heuristic approaches are still signficantly better in most cases, even after 1 hour. Note that as integrality is not strictly enforced on the x-variables, the heuristic approach is not guaranteed to find an integer solution; however, all results obtained were either integer or had an alternative

integer solution with the same objective value. This was verified using a MILP with the only shift types allowed being those found by the heuristic. This MILP takes on average 0.1 seconds to solve. This is a dramatic improvement compared to the full MILP, the results of which are given in Tables 3 and 4.

To provide some more details on how each of the heuristic performs, Table 6 has been included. This reports the maximum number of iterations performed as well as the maximum number of total Benders cuts found by the heuristic approaches on each of the instance sets. From the table it is clearly evident how many fewer iterations can be performed (and hence Benders cuts obtained) by the heuristic on the larger problem classes. Nevertheless, the heuristic is able to find good solutions quickly.

Table 6: Heuristic Statistics - iteration counts and total number of Benders cuts found

	c_0		c_1		c_3		c_5	
Instance Set	it.	cuts	it.	cuts	it.	cuts	it.	cuts
160-1241-600	509	32487	356	45493	276	54908	243	57755
180-1241-600	572	35670	359	45089	268	58097	235	63223
200-1241-600	607	35091	367	46067	296	59842	263	66230
160-1241-1200	982	61584	692	89216	544	110566	464	118429
180-1241-1200	1138	70376	740	91832	548	114855	474	121669
200-1241-1200	1193	69348	723	88664	591	119040	489	122774
160-1241-3600	3082	187326	2027	256277	1676	331539	1378	357015
180-1241-3600	2940	184256	2173	261776	1644	325485	1445	355190
200-1241-3600	3712	213492	2202	258219	1770	346597	1507	369578
160-2145-600	173	10477	137	15171	105	17130	98	18559
180-2145-600	179	10821	135	13915	109	15802	94	17206
200-2145-600	187	11395	117	12164	113	15550	87	16443
160-2145-1200	330	21923	252	28066	201	32452	173	36929
180-2145-1200	348	21244	265	27526	218	32224	179	35259
200-2145-1200	346	20972	231	25006	207	29116	186	32091
160-2145-3600	876	58946	656	74380	600	99495	538	110038
180-2145-3600	922	61888	681	77180	645	98499	544	102234
200-2145-3600	880	56282	664	75151	626	91186	506	99274

To provide an indication of how the approaches behave on specific data sets we have included Figures 2, 3, and 4. Figure 2 provides a comparison of the solution approaches on the 16 different data sets comprising instance set 200-1241-600, while Figure 3 provides a comparison of the solution approaches on the 16 different data sets comprising instance set 200-2145-3600. In both Figures 2 and 3, Gurobi is given the relaxed MILP (1)-(7), and (9), i.e. the integer requirement on the x-variables is removed. Figure 4, on the other hand, gives the same comparison as Figure 3, with the exception that Gurobi is given Model (1)-(9), i.e. integral x-variables as well. All figures display objective function values (in millions) of the best found solution by each approach.

In Figure 2, one can observe the superiority of the heuristic approach on all instances considered. In some cases there is a dramatic difference between the values obtained by the heuristic methods and that of Gurobi. Furthermore, the heuristic approach appears to be more stable in the sense that the objective decreases as the number of allowed shift types

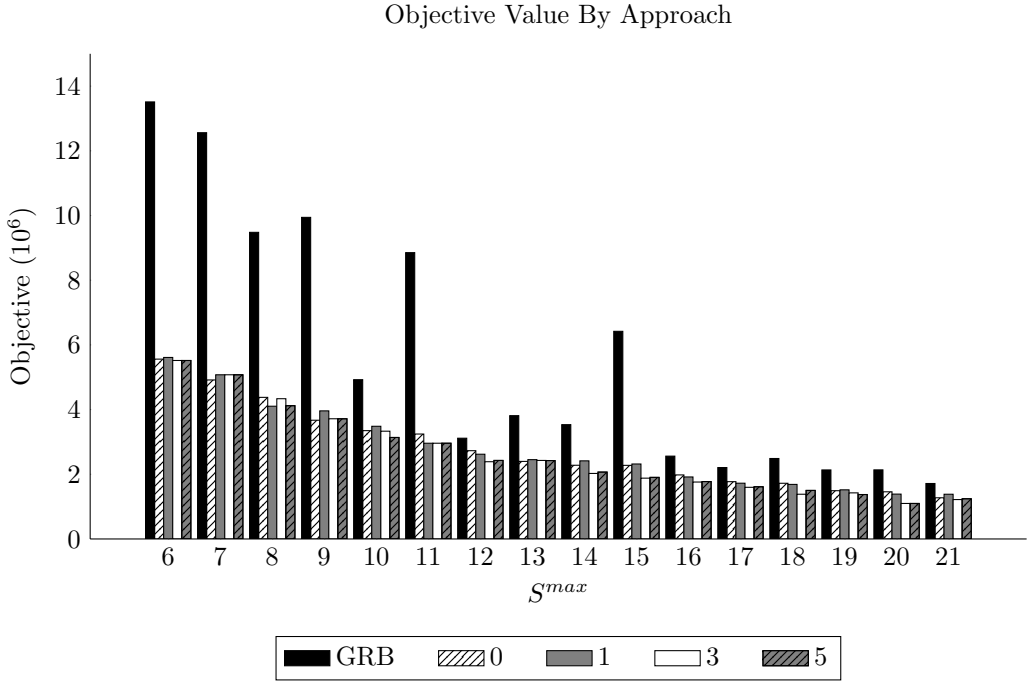


Figure 2: Comparison of solution approaches when $E^{max} = 200, |\mathcal{S}| = 1241$, and a time limit of 600 seconds is enforced. Gurobi is given Model (1)-(7), and (9)

increases. This is inline with intuition; increasing the number of shift types must allow the demand scenarios to be matched more exactly. Gurobi seems to be more erratic in this regard. It is worth mentioning that the best possible solution with 200 available employees and no bound on the number of allowed shift types is 16500. Note that in Figure 2 in particular, and to some extent Figures 3 and 4, the comparison is visually distorted due to the dramatic difference between objective values for some of the instances. Figure 2 is characteristic of problem instances with 1241 shift types and a time limit of 600 seconds. On such instances the heuristic approaches find good solutions quickly.

As expected, Figure 3 indicates a closer comparison between a direct MILP solve (with relaxed x-variables) and the heuristic approaches. Here it would be fair to say that simply using Gurobi outperforms the heuristics when a few number of shift types are allowed. The contrary is true when more shift types are allowed; in some cases improvements of at least 25% can be obtained using the heuristic , see e.g. the CCSDPs with 17 - 20 allowable shift types. This reinforces our conclusion that the Gurobi solve on Model (1)-(7), and (9) is not as stable as the heuristic approaches. Figure 4 shows the poor performance of the full MILP compared to the heuristics.

7. Extensions and Alternative Settings

The problem we present aims at identifying shifts matching several demand scenarios as closely as possible. This is typically done at a tactical level of planning and needs to

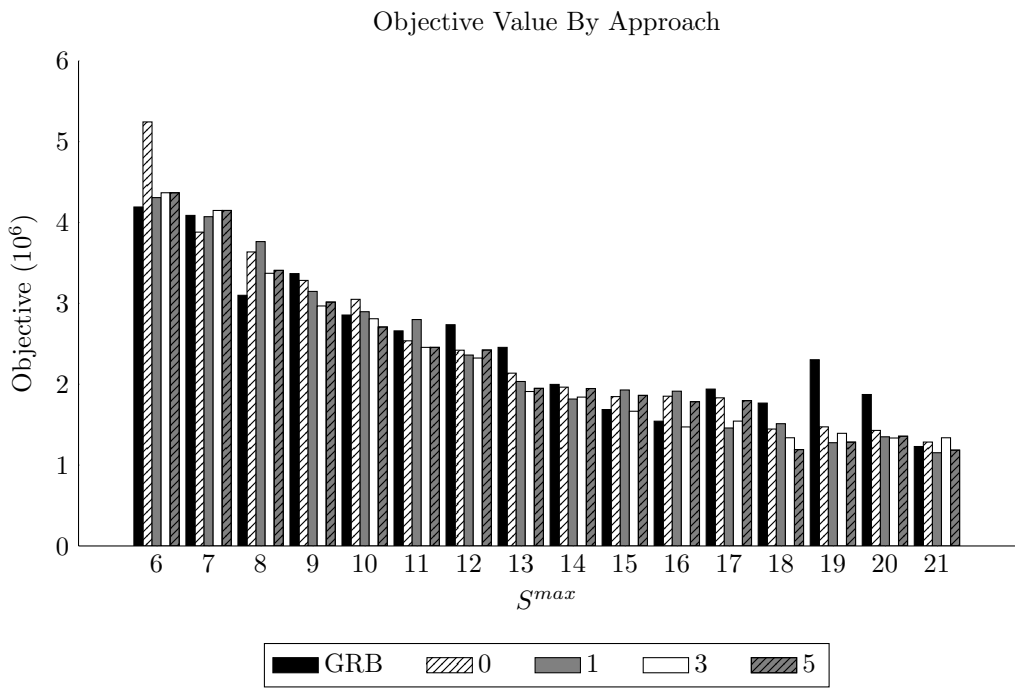


Figure 3: Comparison of solution approaches when $E^{max} = 200, |\mathcal{S}| = 2145$, and a time limit of 3600 seconds is enforced. Gurobi is given Model (1)-(7), and (9)

be reflected in the operational planning of the rostering of shifts according to additional constraints. Lusby et al. (2012) conducts the rostering in accordance to different demand scenarios for a limited number of shift types. They observe that rostering according to demand scenarios improves the match of employees to shifts. A combination of the approach in this paper and the approach described Lusby et al. (2012) may give an even better match.

The approach as presented in this paper assumes that the cost of undercovering and overcovering increases linearly with the increase in undercover or overcover. This assumption may be relaxed by making the cost of e.g. undercover a piecewise linear increasing function of the undercover. In practice one would then add a set of auxiliary variables $0 \leq \overline{u}_{ikt} \leq 1$ with associated cost $c^u_{ikt} \geq 0$ for each k and t and $i = 1, \ldots, d_{kt}$. The cost should be non-decreasing for increasing undercover and therefore $c^u_{i+1kt} \geq c^u_{ikt}$. Similarly we can add auxiliary variables, $0 \leq \overline{o}_{ikt} \leq 1$, for the overcover with non-decreasing costs $c^o_{ikt} \geq 0$. Then

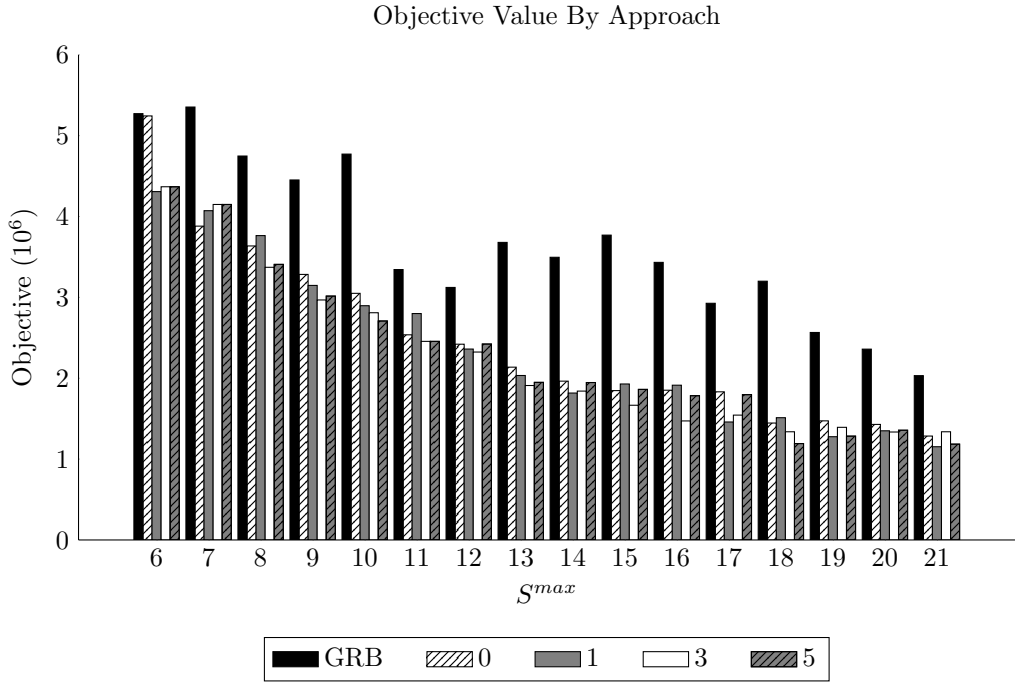


Figure 4: Comparison of solution approaches when $E^{max} = 200, |\mathcal{S}| = 2145$, and a time limit of 3600 seconds is enforced. Gurobi is given Model (1)-(9)

the problem can be reformulated as

$$\min \sum_{k \in \mathcal{K}} \left(\sum_{t \in \mathcal{T}} \sum_{i=1}^{d_{kt}} c^u_{ikt} \overline{u}_{kt} + \sum_{t \in \mathcal{T}} \sum_{i=1}^{d_{kt}} c^o_{ikt} \overline{o}_{kt} \right) \tag{29}$$

$$\text{s.t.} \quad u_{kt} - \sum_{i=1}^{d_{kt}} \overline{u}_{kt} = 0, \qquad k \in \mathcal{K}, t \in \mathcal{T} \tag{30}$$

$$o_{kt} - \sum_{i=1}^{d_{kt}} \overline{o}_{kt} = 0, \qquad k \in \mathcal{K}, t \in \mathcal{T} \tag{31}$$

$$(2) - (9)$$

$$0 \leq \overline{u}_{ikt} \leq 1, \quad k \in \mathcal{K}, t \in \mathcal{T}, i = 1, \ldots, d_{kt} \tag{32}$$

$$0 \leq \overline{o}_{ikt} \leq 1, \quad k \in \mathcal{K}, t \in \mathcal{T}, i = 1, \ldots, d_{kt} \tag{33}$$

Thus a higher deviation from the projected demand the relative higher cost it will have. As a consequence the model will limit the number of cases where one demand scenario will have almost perfect cover while another will have a large deviation from the projected demand.

The addition of constraints (30) and (31) will not change the shape of the Benders cuts (22) as the right-hand-side coefficients of these constraints are zero. The upper bounds on the auxiliary variables will, however, add to the constant term of the Benders cuts. The most significant change is for the model (17)-(21) and as a consequence requires a modification of

the search for dual alternative solutions. We have left this modification for future research.

A related problem is when we are interested in making the worst case scenario as good as possible. That is, instead of summarizing the objectives of the subproblems we would like to minimize the worst objective. This can be accommodated by a slight change of the model by adding the constraint

$$z \geq \sum_{t \in \mathcal{T}} c^u u_{kt} + \sum_{t \in \mathcal{T}} c^o o_{kt} \tag{34}$$

and then minimize z in the objective. The Benders cuts will be changed slightly as all cuts from all scenarios will use z instead of z_k in (22), while the objective (23) just has to minimize z and not the sum of z_k.

An implicit assumption of the model is that a solution having neither undercover nor overcover of any periods is the one we strive for. As a consequence of this we assume that the cost of covering the actual demand is accepted and is sunk. Hence we do not minimize the cost of actual staffing levels. In our model we have a maximum number of employees who can be allocated to shifts in each scenario. While this upper bound may give an upper bound on the available time one might instead set bounds on the number of working hours available in each of the scenarios e.g. +/- 10% of the total time demanded. Narrowing the upper bound will limit the staffing cost but at the cost of potential under and over cost. This change may introduce more fractionality into the subproblems, as it is ruins the unimodular structure, which then has to be handled.

Finally, in practice, it is often better to undercover periods where the demand is high compared to periods where the demand is low. The reason is that undercovering by a single employee in a high demand period is a relatively small undercoverage compared to an undercover by a single employee in a low demand period. This can in practice be emphasized in the model with the piecewise linear objective presented above.

8. Conclusion

In this paper we have formulated the CCSDP as a MILP that lends itself very naturally to Benders Decomposition and described a special case of the SDP that is easy to solve. Due to convergence issues with standard Benders decomposition, we have proposed a Benders decomposition based matheuristic for solving this problem. Furthermore, we have outlined a technique for finding dual alternative optimal solutions to the Benders subproblem; each dual alternative optimal solution can be used to construct a Benders cut and thus provide information for the Benders master problem.

The proposed heuristic considers a small set of shift types at any one time and is dynamically updated upon analysis of the Benders cuts obtained to find the optimal solution to the reduced problem. Improving shift types are found by looking at the coefficients of the Benders cuts and identifying those which could potentially improve the solution quality.

Computational experiments performed on 96 test instances based on realistic data provided by Copenhagen Airport confirm the efficiency of the proposed methodology. On instances containing 1241 shift types the proposed heuristic outperformed a direct MILP solve using Gurobi, providing solutions that were on average 40% better when a time limit of 600 seconds was enforced and solutions that were around 25% better over 20 minutes. Furthermore, the heuristic approach remains competitive when a one hour time limit is observed, yielding an improvement over the MILP of around 3-18%. Looking at the results,

the heuristic appears to be more stable than the MILP solve. Finding dual alternative optimal solutions also seems to improve the performance of the heuristic, often finding better solutions than when it was not used. This is despite the fact that searching alternative solutions reduces the number of solutions to a CCSDP the algorithm can consider, given a certain time limit.

On instances with 2145 shift types the results are not so definitive; directly solving the MILP outperforms the Benders heuristic when few shift types are allowed, but is unable to compete with the heuristic when the number of shift types increases. On average, however, the heuristic approaches find solutions that are 10-19% better within 10 minutes, 7-15% better over 20 minutes, and 2-11% better after one hour.

Interesting extensions of the current work include speeding up the Benders decomposition component of the algorithm. Despite the fact that we only ever consider a severely limited set of shift types, this can be cumbersome at times. The dramatic difference in the number of iterations the heuristic can perform on each of the two data sets is evidence to support this. Furthermore, the proposed framework can easily be applied in a stochastic programming environment, where one does not have a daily demand scenario for each day of the week (as is the case here) but multiple demand scenarios (each with a given probability of occuring) for a given day.

References

References

Aykin, T., 1996. Optimal shift scheduling with multiple break windows. Management Science 42 (4), 591–602.

Aykin, T., 1998. A composite branch and cut algorithm for optimal shift scheduling with multiple breaks and break windows. Journal of the Operational Research Society 49 (6), 603–615.

Aykin, T., 2000. A comparative evaluation of modeling approaches to the labor shift scheduling problem. European Journal of Operational Research 125 (2), 381–397.

Bechtold, S., Jacobs, L., 1990. Implicit modeling of flexible break assignments in optimal shift scheduling. Management Science 36 (11), 1339–1351.

Benders, J., 1962. Partitioning procedures for solving mixed-variables programming problems. Numerische Mathematik 4 (1), 238–252.

Butler, D., Maydell, U., 1979. Manpower scheduling in the Edmonton police department. INFOR Journal 17 (4), 366 – 372.

Cordeau, J., Stojkovic, G., Soumis, F., Desrosiers, J., 2001. Benders decomposition for simultaneous aircraft routing and crew scheduling. Transportation Science 35 (4), 375–388.

Dantzig, G. B., 1954. Letter to the editora comment on edie's traffic delays at toll booths? Journal of the Operations Research Society of America 2 (3), 339–341.

Di Gaspero, L., Gärtner, J., Kortsarz, G., Musliu, N., Schaerf, A., Slany, W., 2007. The minimum shift design problem. Annals of Operations Research 155 (1), 79–105.

Dirickx, Y., Jennegren, L., 1979. Systems analysis by multi-level methods: with applications to economics and management. John Wiley & Sons, Inc.

Ernst, A., Jiang, H., Krishnamoorthy, M., Owens, B., Sier, D., 2004a. An annotated bibliography of personnel scheduling and rostering. Annals of Operations Research 127 (1-4), 21–144.

Ernst, A., Jiang, H., Krishnamoorthy, M., Sier, D., 2004b. Staff scheduling and rostering: A review of applications, methods and models. European Journal of Operational Research 153 (1), 3–27.

Eveborn, P., Rönnqvist, M., 2004. Scheduler - a system for staff planning. Annals of Operations Research 128 (1-4), 21–45, cited By 20.

Gaballa, A., Pearce, W., 1979. Telephone sales manpower-planning at qantas. Interfaces 9 (3), 1–9.

Geoffrion, A., 1972. Generalized benders decomposition. Journal of Optimization Theory and Applications 10 (4), 237–260.

Glover, F., Glover, R., McMillan, C., 1983. Heuristic programming approach to the employee scheduling problem and some thoughts on 'managerial robots'. Proceedings of the Hawaii International Conference on System Science, 420–436.

Lusby, R., Dohn, A., Range, T. M., Larsen, J., 2012. A column generation-based heuristic for rostering with work patterns. Journal of the Operational Research Society 63, 261–277.

Magnanti, T., Wong, R., 1981. Accelerating benders decomposition - algorithmic enhancement and model selection criteria. Operations Research 29 (3), 464–484.

Mason, A. J., Ryan, D. M., Panton, D. M., 1998. Integrated simulation, heuristic and optimisation approaches to staff scheduling. Operations Research 46 (2), 161–175.

Mehrotra, A., Murphy, K., Trick, M., 2000. Optimal shift scheduling: A branch-and-price approach. Naval Research Logistics 47 (3), 185–200.

Musliu, N., Schaerf, A., Slany, W., 2004. Local search for shift design. European Journal of Operational Research 153 (1), 51 – 64, timetabling and Rostering.

Nielsen, S. B., September 2012. Staff scheduling at copenhagen airport. Master's thesis, Department of Management Engineering, Technical University of Denmark.

Rekik, M., Cordeau, J., Soumis, F., 2004. Using benders decomposition to implicitly model tour scheduling. Annals of Operations Research 128 (1-4), 111–133.

Rousseau, L.-M., Gendreau, M., Feillet, D., 2007. Interior point stabilization for column generation. Operations Research Letters 35 (5), 660–668.

Schrijver, A., 1986. Theory of Linear and Integer Programming. John Wiley & Sons, Inc., New York, NY, USA.

Segal, M., 1974. The operator-scheduling problem: A network-flow approach. Operations Research 22 (4), 808 – 823.

Thompson, G., 1990. Shift scheduling in services when employees have limited availability: An l.p. approach. Journal of Operations Management 9 (3), 352–370.

Thompson, G., 1996. A simulated-annealing heuristic for shift scheduling using non-continuously available employees. Computers and Operations Research 23 (3), 275–288.

Van den Bergh, J., Belien, J., De Bruecker, P., Demeulemeester, E., De Boeck, L., 2013. Personnel scheduling: A literature review. European Journal of Operational Research 226 (3), 367–385.